What is claimed is:

(Claim 1) *1*.A method for identifying return instruction, comprising:(a) providing a return target stack at initialization;

(b) fetching a current instruction;
(c) if said current instruction being a call instruction, adding an address of said current instruction with a length of said current instruction for obtaining a target pointer to store in said return target stack; and
(d) if an address of a subsequent instruction after said current instruction is executed being identical to said target pointer stored in said return target stack, said current instruction is identified as a return instruction.

(Claim 2) *2*.The method as recited in claim *1*, further comprising:providing a return instruction address table; and

storing an address of said return instruction to said return instruction address table.

(Claim 3) *3*.The method as recited in claim *2*, further comprising:deleting said target pointer identical to said subsequent instruction from said return target stack.

(Claim 4) *4*.The method as recited in claim *1*, further comprising:providing a plurality of rows to said return target stack;

providing an effective flag corresponding to said rows respectively; and
when said target pointer being stored to one of said rows, setting said effective flag corresponding to said row which stores said target pointer.

(Claim 5) *5*.The method as recited in claim *4*, further comprising:clearing said effective flag corresponding to each of said rows respectively at initialization.

(Claim 6) *6.*The method as recited in claim *1*, wherein said return target stack is a circular queue.

(Claim 7) *7.*A method for predicting target pointer, comprising:(a) providing a return target stack and a return instruction address at initialization;

(b) fetching a current instruction;
(c) if said current instruction being a call instruction, adding an address of said current instruction with an length of said current instruction for obtaining a target pointer to store in said return target stack;
(d) if an address of a subsequent instruction after said current instruction is executed being identical to said target pointer stored in said return target stack, said current instruction is identified as a return instruction;
(e) if said current instruction is identified as a return instruction, an address of said return instruction being stored in said return instruction address table;
(f) if said current instruction is identified as a return instruction, deleting said target pointer identical to said subsequent instruction from said return target stack; and
(g) if an address of said current instruction being stored in said return instruction address table, an address on topmost layer of said return target layer is assigned as an address of a next instruction.

(Claim 8) *8.*The method as recited in claim *7*, further comprising:providing a plurality of rows to said return target stack;

providing an effective flag corresponding to each of said rows respectively; and
when said target pointer being stored to one of said rows, setting said effective flag corresponding to said row which stores said target pointer.

(Claim 9) *9.*The method as recited in claim *8*, further comprising:clearing said effective flag corresponding to each of said rows respectively at initialization.

(Claim 10) *10.*The method as recited in claim *7*, further comprising:providing a plurality of rows to said return instruction address table;

providing an effective flag corresponding to each of said rows respectively; and

when said target pointer being stored to one of said rows, setting said effective flags corresponding to said row which stores said target pointer.

(Claim 11) *11.*The method as recited in claim *10*, further comprising:clearing said effective flags corresponding to respective said rows at initialization.

(Claim 12) *12.*The method as recited in claim *7*, wherein said return target stack is a circular queue.